

What is claimed is:

1. A method for enabling an application designer to develop user interfaces (UIs) by modeling without coding, the method comprising the steps of:

- a. identifying the requirements of the UI as processes, the application designer identifying the processes;
- b. defining the tasks that are required to define the identified processes, the tasks being defined by providing meta data to the instances of a set of pre-built reusable components, each pre-built reusable component being an abstract object built to perform a function;
- c. connecting the tasks in a logical order to model the processes, the tasks being connected in a flowchart like manner; and
- d. storing the data related to the tasks and the overall process for the developed UI in a database.

2. The method according to claim 1, wherein the step of identifying the processes comprises the steps of:

- a. identifying the processes required to develop the UI screens; and
- b. identifying the processes required to develop the application controller for the UI.

3. The method according to claim 1, wherein the step of defining the tasks comprises the steps of:

- a. defining the tasks that are required to define the processes that are required to develop the UI screens comprising:
  - i. defining the tasks for placing controls on the UI screens;
  - ii. defining the tasks for mapping of database fields to the screen variables;

- iii. defining the tasks for validating the user actions at the control level and at the screen level;
    - iv. defining the tasks for forwarding the screen information to the application controller; and
  - 5        b. defining the tasks that are required to define the processes that are required to develop the application controller for the UI.
- 4. The method according to claim 3, wherein the step of defining the tasks comprises the steps of:
  - 10        a. selecting the components that are required to define the identified tasks; and
  - b. defining the tasks, the tasks being defined by specifying meta-data for the selected components, the meta-data being process specific properties which when associated with a component defines the corresponding task.
- 15        5. The method according to claim 1, wherein the step of interconnecting the tasks comprises the steps of:
  - a. defining at least one Rule;
  - b. specifying a task to be referred to in case of success of the Rule; and
  - c. specifying a task to be referred to in case of failure of the Rule.
- 20        6. The method according to claim 1, wherein the step of interconnecting the tasks comprises the step of connecting the tasks serially.
- 7. The method according to claim 1, further comprises the step of enabling the application designer to visually verify the developed UI, the verification comprising the steps of:
  - a. adding a plurality of breakpoints in the UI process;

- b. validating the UI process till the first breakpoint;
- c. presenting the results of verification to the application designer; and
- d. verifying the remaining part of the UI process.

8. The method according to claim 7, wherein the step of verifying the UI is performed until the next breakpoint in the UI.

9. The method according to claim 7, wherein the step of verifying the UI is performed for the remaining part of the UI ignoring the breakpoints.

10. The method according to claim 7, wherein verifying the UI is performed task by task for the remaining part of the UI.

11. The method according to claim 1 further comprises the step of enabling a user to execute the developed UI, the execution comprising the steps of:

- a. inputting a request to be processed, the request being input by a user;
- b. transferring the request to an Engine for processing;
- c. identifying the components required to process the user's request;
- d. caching the identified components;
- e. executing the tasks in a logical order as defined in the UI process;
- f. handling the errors that occur while processing the request, if errors occur in any of the above steps;
- g. logging the information related to the execution of the tasks in a database; and
- h. outputting the results of the execution.

12. A method for enabling an application designer to develop User Interfaces (UIs) by modeling without coding, the method comprising the steps of:

- a. identifying the processes that are required to model the UI screens, the application designer identifying the processes;
- b. identifying the processes that are required to develop the application controller for the UI;
- 5 c. defining the tasks that are required to define the identified processes, the tasks being defined using a set of pre-built reusable components, each pre-built reusable component being an abstract object built to perform a function;
- d. defining a plurality of Rules;
- 10 e. specifying a task to be referred to in case of the success of each Rule;
- f. specifying a task to be referred to in case of a failure of each Rule;
- g. interconnecting the tasks serially;
- h. visually verifying the developed UI; and
- i. storing the data related to the developed UI in a database.
- 15 13. A system for enabling an application designer to develop user interfaces (UIs) by modeling without coding, the system comprising:
  - a. an Engine, the Engine executing the UIs;
  - b. a plurality of clients, the clients interacting with the Engine;
  - 20 c. a Designer for providing an interface to the application designer to visually develop and verify the UIs, the Designer being one of the plurality of the clients; and
  - d. a plurality of Databases for storing meta-data and application data related to the UIs modeled using the Designer.

14. The system according to claim 13, further comprising:

- a. a Web Server, the Web Server connecting the Engine with the plurality of clients;
- b. an Administration Tool, the Administration Tool enabling a user to view the system activity and perform the administrative functions, the Administration Tool being one of the plurality of clients;
- c. a Message Service module, the Message Service module enabling the interaction of the Engine with the plurality of clients; and
- d. A Security Service module, the Security Service module providing the authentication, authorization, auditing and administration services.

15. The system according to claim 13, wherein the Engine comprises:

- a. a Message Service Queue, the Message Service Queue temporarily storing requests and the results obtained by processing of the requests, the requests being processed by the Engine;
- b. a Scheduler, the Scheduler scheduling the requests into the Message Service Queue;
- c. a DB Logger, the DB Logger logging the information related to the requests in the Database;
- d. a Cache Manager, the Cache Manager caching meta-data related to the business application models from the Database, the business application model being executed for processing the request;
- e. an Execution Module, the Execution Module executing the requests; and
- f. an Exception Handling Framework, the Exception Handling Framework for handling the errors that occur during processing of the requests.

16. The system according to claim 13 further comprising a plurality of pre-built components, the components enabling the application designer to develop the UIs, each component representing a functionality that is required in the development of UIs, each component comprising:

- 5           a. a meta-data structure that defines the structure of the meta-data, wherein the meta-data is a collection of properties specific to the functionality being represented by the component;
- 10           b. a plurality of GUI objects that encapsulate the meta-data structure and provide a GUI to enable the application designer to input the meta-data; and
- c. a computer program code segment that understands the meta-data structure and the implied functionality.

17. The system according to claim 16, wherein the plurality of pre-built components comprise Functional components, the Functional components comprising:

- 15           a. a UI State component representing the functionality of specifying the system-defined state and User defined state for the UI;
- b. a Database Access component representing the functionality of accessing data from a Database;
- 20           c. an External Object component representing the functionality of using an external object;
- d. a Rule component representing the functionality of evaluation of a condition; and
- e. an Assignment Object representing the functionality of assigning a value to the variables used in the UI.

18. The system according to claim 16, wherein the plurality of pre-built components comprise Flow components, the Flow components comprising:

- a. a Selector component representing the functionality of selecting one of the components in the UI;
- 5 b. a UI Start component representing the functionality of denoting where a UI starts;
- c. a UI End component representing the functionality of denoting where a UI ends;
- 10 d. a Line component representing the functionality of connecting the components;
- e. a Rule component representing the functionality of evaluation of a condition;
- f. a Tasks component representing the functionality of calling the Rule component;
- 15 g. a Loop component representing the functionality of modeling repetitive execution of tasks within a process;
- h. a Loop end component representing the functionality of terminating a loop within a process; and
- i. a Text component representing the functionality of annotation.

20 19. The system according to claim 13, wherein the Designer comprises:

- a. A UI modeling screen for enabling the application designer to model the UI in a flowchart like manner;
- b. a UI Start component's task screen for enabling the application designer to specify the meta-data for the UI start component;

- c. a Line component's task screen for enabling the application designer to specify the meta-data for the Line component;
- d. a Rule component's task screen for enabling the application designer to specify the meta-data for the Rule component;
- 5 e. a Rule condition screen for enabling the application designer to specify the left hand side and right hand side of the Rule component;
- f. a Process State component's task screen for enabling the application designer to specify the meta-data for the Process State component;
- 10 g. an Assignment Object's task screen for enabling the application designer to specify the meta-data for the Assignment component; and
- h. Database Access component's task screen for enabling the application designer to specify the meta-data for the Database Access component.

20. The system according to claim 19, wherein the UI modeling screen comprises:

- a. a UI Workspace for defining the UI in a flowchart like manner;
- 15 b. a Component Palette for providing an interface to the components for defining the UI; and
- c. an Object Browser for providing an interface to the components for defining the UI.

20 21. The system according to claim 15, wherein the Exception Handling Framework comprises:

- a. an Error Types and Structure, the Error Types and Structure storing information regarding all the possible errors;
- b. an Exception Handling Strategy, the Exception Handling Strategy identifying and executing a strategy for handling the errors;



- c. an Error Integration, the Error Integration comprising a library of strategies for handling the errors, the library being integrated with the Exception Handling Strategy; and
- d. an Error Logger, the Error Logger storing the information related to the errors in the Database.

22. The system according to claim 21, wherein the Error Types and Structure comprises:

- a. an Error Object, the Error Object encapsulating all the information about an error and its context; and
- b. an Exception Hierarchy, the Exception Hierarchy maintaining a list of all errors grouped in a hierarchical structure.

23. The system according to claim 22, wherein the Exception Handling Strategy comprises:

- a. an Error Handler, the Error Handler executing the strategy for handling the errors;
- b. an Error Default Handler, the Error Default Handler specifying the default strategies for handling any unhandled exceptions;
- c. a Resource Preallocation, the Resource Preallocation managing the memory that stores the information related to the errors; and
- d. a plurality of Error dialogues, the Error dialogues displaying an error message to the user.

24. A computer program product for use with a computer, the computer program product comprising a computer usable medium having a computer readable program code embodied therein for enabling a designer to model User interfaces by modeling without coding, the computer program code performing the steps of:

- 5
- a. identifying the processes that are required to model the UI, the application designer identifying the processes;
  - b. defining the tasks that are required to define the identified processes, the tasks being defined using a set of pre-built reusable components, each pre-built reusable component being an abstract object built to perform a function;
  - c. connecting the tasks in a logical order to model the processes, the tasks being connected in a flowchart like manner;
  - d. storing the data related to the developed UI in a database.
- 10
25. The computer program product as claimed in claim 24, the computer program product further comprising program code embodied therein for enabling a user to visually verify the developed UIs.
- 15
26. A computer program product as claimed in claim 24, the computer program product further comprising program code embodied therein for enabling a user to execute the user interface using the computer program product, the computer program code performing the steps of:
- a. inputting a request to be processed, the request being input by a user;
  - b. transferring the request to an Engine for processing;
  - c. identifying the components required to process the user's request;
  - d. caching the identified components;
  - e. executing the tasks in a logical order as defined in the UI process;
  - f. handling the errors that occur while processing the request, if errors occur in any of the above steps;
- 20

- g. logging the information related to the execution of the tasks in the Database.